Amendments to the Claims:

This listing of claims will replace all prior versions, and listings of claims in the application:

Listing of Claims:

1-23. (Canceled)

24. (New) A system for adapting foreign binary code compiled for a foreign computer system for use on a native computer system, the system comprising:

a central processing unit (CPU);

a volatile memory, wherein at least a portion of the volatile memory is concealed from memory access transactions initiated by the foreign binary code;

a non-volatile memory, wherein at least a portion of the non-volatile memory is concealed from storage access transactions initiated by the foreign binary code; and

a translator configured to translate the foreign binary code into a translated code, wherein the CPU is capable of executing the translated code on the native computer system, the translator further configured to store the translated code in either the volatile memory or the non-volatile memory or both.

25. (New) The system of claim 24 wherein the translator is further configured to store a first portion of the translated code in the non-volatile memory and a second portion of the translated code in the volatile memory.

26. (New) The system of claim 25 wherein the first portion of the translated code stored in the non-volatile memory is optimized.

27. (New) The system of claim 24 wherein the volatile memory includes a random access memory.

28. (New) A method for improving execution performance of foreign binary code on a native computer system, the method comprising:

managing an association between the foreign binary code and a translated code, wherein the translated code is stored in either a volatile memory or a non-volatile memory;

identifying a portion of the translated code that is associated with a corresponding portion of the foreign binary code from either the volatile memory or the non-volatile memory;

if the identified portion of the translated code is stored in the non-volatile memory, transferring the identified portion of the translated code from the non-volatile memory to the volatile memory without significant modifications;

performing a set of optimizing transformations on a portion of the translated code stored in the volatile memory to generate a transformed portion of the translated code, wherein the transformed portion includes improved execution performance characteristics; and

storing the transformed portion in the non-volatile memory.

29 (New) A binary translation system for use with a native computer system to provide software compatibility with a foreign computer system, the binary translation system comprising:

dynamic binary translator software comprising:

a plurality of binary translators, each binary translator configured to optimize a translated binary code at a designated optimization level; and

a plurality of dynamic support routines for use by the plurality of binary translators;

a code cache configured to store a most recently used portion of a translated binary code, wherein the translated binary code corresponds to a foreign binary code compiled for use with the foreign computer system, and wherein the code cache is implemented using a portion of a volatile memory of the native computer system, the code cache being available only to the binary translation system and the translated binary code;

first set of logic configured to retrieve contents from the code cache in response to corresponding foreign code branch events;

a non-volatile database configured to store optimized portions of the translated binary code; and

second set of logic configured to retrieve contents from the non-volatile database in response to corresponding portions of the foreign binary code being discovered in the volatile memory of the native computer system and transfer retrieved contents to the code cache.

30. (New) The binary translation system of claim 29, wherein the non-volatile database is concealed from the foreign binary code by means of a separate external storage device attached to a peripheral channel of the native computer system.

31. (New) The binary translation system of claim 29, wherein the non-volatile database is concealed from the foreign binary code by means of a peripheral access filtering mechanism;

wherein accesses to the non-volatile database from the foreign binary code are analyzed by the binary translation system;

wherein accesses are allowed if such accesses target the foreign binary code and any accompanying data sets;

wherein accesses are disallowed if such accesses target storage space occupied by the non-volatile database; and

wherein overall capacity and related parameters of the non-volatile database attributed to the foreign binary code are corrected to account for loss of space in the same way.

32. (New) The binary translation system of claim 29, wherein the second set of logic is further configured to:

intercept a foreign code-initiated external storage access operation by the binary translation system;

maintain an association structure, filled to provide associations between a computer system memory address group and an external storage address group, the external storage address group expressed in terms of logical blocks or any other suitable entities;

for any storage-to-memory read operations, update the association structure to note which memory addresses get loaded from which logical blocks, wherein the loaded memory

addresses are subsequently monitored against future modification attempts by the foreign binary code;

for any memory-to-storage write operations, update the association structure to note which memory addresses get stored to which logical blocks, wherein the memory addresses are subsequently monitored against future modification attempts by the foreign binary code, and search the non-volatile database for optimized translated binary code sequences associated with now overwritten logical blocks and invalidate any such sequences;

for any modification attempt on the monitored memory addresses, perform the invalidation of the corresponding association;

for optimized translated binary code sequences saved into the non-volatile database, save the associations between the foreign binary code address groups and the corresponding logical blocks, thereby associating the optimized translated binary code sequences with a group of logical blocks;

should a need to search for an optimized translated binary code for a group of foreign binary code memory addresses arise, check an association table to find which logical blocks supplied the foreign binary code that is now available at the corresponding foreign binary code memory addresses;

should an association be discovered in the association table, search the non-volatile database to locate the optimized translated binary code sequences associated with the association table-reported logical blocks numbers; and

should a sequence be successfully discovered in the database, load the corresponding optimized translated binary code from the database into the code cache and attach the loaded code to address mapping facilities, thereby directing all future execution of the foreign binary code at the specified foreign address to the optimized translated binary code.

33. (New) The binary translation system of claim 32, wherein the second set of logic is further configured to:

store an image of the foreign binary code translated into optimized translated binary code sequences together with the aforementioned sequences into th database;

associate each optimized binary translated sequence with the corresponding foreign code image;

should a need to search for an optimized translated binary code for a group of foreign code memory addresses arise, search the database for the foreign code image identical to the actual contents of the memory at the memory addresses;

should such an image be successfully located in the database, load the corresponding optimized translate binary code from the database into the code cache and attach the loaded code to address mapping facilities, thereby directing all future execution of the code at the specified foreign address to the optimized translated binary code.

34. (New) The binary translation system of claim 33, wherein the second set of logic is further configured to:

calculate a set of uniform identification values for every foreign code image being stored to the database, said set of values calculated in such a way as to provide a digest of contents of the image with a relatively low collision rate, said digests being smaller, faster and easier to effectively compare than the foreign code images;

associate the digests with the foreign code images;

should a need to search for an optimized translated binary code for a group of foreign code memory addresses arise, calculate the digest for the actual contents of the memory at the memory addresses;

compare the calculated digest against all the digests for the images stored in the database to identify a set of images whose contents produce the same digest;

for each image in the identified set of images, order a complete image comparison to find an image identical to the actual contents of the memory at the memory addresses; and

should such an image be successfully discovered in the database, load the corresponding optimized translated binary code from the database into the code cache and attach the loaded code address mapping facilities, thereby directing all future execution of the code at the specified foreign address to the optimized translated binary code.

35. (New) The binary translation system of claim 34, wherein should the search in the association structure fail to locate any association between the foreign memory addresses and the logical blocks, or should the search in the database fail to locate any optimized binary translated codes associated with the source storage blocks, the second set of logic is engaged to perform database retrieval.